

## EJERCICIO 1:

Primero quiero notar cuantos dígitos necesito para contar un número en una base cualquiera. Por ejemplo base tres con un dígito se puede contar hasta 1

Si  $3 \neq 0$  obviamente

Notemos dos cosas: (1)  $\forall x, y$  existen  $i, i+1$  tal que

$3^{i+1} \geq x \geq 3^i$ , o sea podemos "atrapar" a  $x$  entre dos potencias.

(2) Para toda base  $K$  con hasta  $j$  dígitos se puede contar

hasta  $K^j - 1$ , pues  $K^{j+1} = \underbrace{10000}_{j \text{ dígitos}} = 1$

O sea el mínimo número en base  $K$  que requiere  $j+1$  dígitos es  $K^{j+1}$  y el mayor número en base  $K$  que requiere  $j+1$  dígitos es  $K^{j+1} - 1$ , pues  $K^{j+2}$  es el mínimo que requiere  $j+2$ .

Entonces basta con saber cuál es la mínima potencia  $3^j$  tal que  $3^j > x$

LUEGO

$$f(x, y) = \underbrace{x \cdot (y=1)}_{\text{Prueba es } P_r} + \underbrace{(x+1) \cdot (y=0)}_{\text{Suma es } P_r} + \min_{t \leq x} \left( (3^t > x) \wedge y > 1 \right) \quad \text{MINIMIZACIÓN ACOTADA ES } P_r$$

LUEGO ESTO ES  $P_r$  PORQUE:  $\leq$   $\geq$  ES  $P_r$

Todo lo celeste de arriba.

La parte no hace falta escribirlo así, se puede notar con  $\{$  que es más claro y ya sabemos  $1 \leq 1 \leq P_r$ .

3

BOCACIOS JERARQUÍA

EJERCICIO 2

59

SUPONGAMOS  $g$  ES COMPUTABLE. TAMBIÉN LO ES

~~$g'(x) = g(x)$~~

$$g'(x) = g(x)$$

$$\text{SEA } h(x) = \begin{cases} 1 & \text{SI } g'(x) \\ \text{X} & \text{SI NO.} \end{cases}$$

<sup>total</sup>  $h$  ES COMPUTABLE POR SU DEFINICIÓN DE CASOS MUTUAMENTE EXCLUSIVOS

ENTONCES POR TEOREMA DE LA RECURSIVIDAD EXISTE  $e$  TAL QUE

$$\phi_e(y) = h(y)$$

$$\phi_e(e) = \begin{cases} 1 & \text{SI } \phi_e(e) \downarrow \text{ Y } e \leq \phi_e(e) \leq ze \\ \text{X} & \text{SI NO} \end{cases}$$

- SUPONGAMOS QUE  $\phi_e(e) = 1 \Rightarrow \phi_e(e) \downarrow$  Y  $e \leq \phi_e(e) \leq ze$ . PERO  $e$  NO ES NI 1 NI <sup>total</sup> UNOS LOS PROGRAMAS QUE CODIFICA NÚMERO UNO UNO ES ESTE. ENTONCES NO PASA QUE  $e \leq 1 \leq ze$

- SUPONGAMOS  $\phi_e(e) = ze \Rightarrow \phi_e(e) \uparrow$  O  $\phi_e(e)$ . ~~NO PUEDE SER MÁS GRANDE QUE  $ze$  PORQUE  $ze$  ES EL MÁXIMO VALOR QUE PUEDE TOMAR  $\phi_e$~~   $\phi_e(e) \uparrow$  Y  $\neg(e \leq \phi_e(e) \leq ze)$

LUEGO COMO LO ÚNICO QUE HABÍAMOS SUPUESTO ERA QUE  $g$  ERA COMPUTABLE. ESA SUPOSICIÓN ERA FALSA Y  $g$  NO ERA COMPUTABLE.

ENTONCES  $g'(e)$  NO VALE, COMO  $\phi_e(e)$  SÓLO PUEDE SER QUE  $\phi_e(e)$  NO ESTE EN EL RANGO DE VALORES, PERO  $e \leq \phi_e(e) \leq ze$ . ABS!



10

9

① ②

$$\phi(e) = \begin{cases} \uparrow & \text{si } e \in \text{DOM}(\phi_e) \\ e & \text{si no} \end{cases}$$

- SUPONAMOS QUE  $e \in \text{DOM}(\phi_e) \Rightarrow \phi(e) \downarrow$  PERO  $\phi(e) \neq \uparrow$
- SUPONAMOS QUE  $e \notin \text{DOM}(\phi_e) \Rightarrow \phi(e) \uparrow$ , PERO  $\phi(e) = e$

ABSÓ

LUEGO COMO LO ÚNICO QUE HABÍAMOS SUPUESTO ES A COMPUTABLE, SUCEDE QUE A ES NO COMPUTABLE  $\Rightarrow$  A NO ES PR

—  
SABEMOS QUE UNA FUNCIÓN ES C.E. SI ES EL DOMINIO DE UNA FUNCIÓN PARCIAL COMPUTABLE.

$$\text{SEA } f: K \rightarrow \{0,1\} \text{ SEA } f(x,y,z) = \min_e ( \text{STP}(T(T(K)), R(K); t) = 1 ) \wedge \text{STP}(R(K); t) = 1 )$$

NOTAMOS QUE ESTE PROGRAMA TERMINA SI  $\phi_x(z) \downarrow$  Y  $\phi_y(z) \downarrow$  Y SE DETERMINA SU VALOR.

f ES PARCIAL COMPUTABLE PUES ES UNA MINIMIZACIÓN NO ACOTADA. LUEGO POR LO DE ARRIBA A ES C.E.

LUEGO A ME ES C.E. PORQUE SI NO FUESE SERÍA COMPUTABLE, QUE NO LO ES. C.E. A?



## EJERCICIO 4:

a)

PARA PROBARLO VOY A ESCRIBIR UN PROGRAMA EN S QUE LO COMPUTA.

SUPONGO QUE A MI PROGRAMA LO ENTRA X POR  $X_1$  Y K POR  $X_2$ .

→ PRACTICA 2 EJ 1A

 $Z_3 \leftarrow 0$  // K-enchado en valor X $Z_4 \leftarrow 0$  // GUARDE ACA SI TERMINA O NOA  $Z_4 \leftarrow \text{STP}(Z_3, X_1, X_2)$  → STP ES P.E) COMPUTABLE → QUE HAY UNA MÁQUINA QUE LO COMPUTAIF  $Z_4 = 0$  GO TO P

GO TO I → PRACTICA 2 EJ 1

I  $Z_3 \leftarrow 0$  $Z_3 \leftarrow Z_3 + 1$ 

GO TO A

P  $Y \leftarrow Y + 1$  → GO TO E

(HACIA QUE FORMAR CUIDADO NO SE PUEEN SELECCIONAR, SUPONGO QUE NO SE PUEEN ENTRE MÁQUINAS.)

EL PROGRAMA VA EVALUANDO  $\text{STP}(Y, X, K)$  PARA TODOS LOS Y HASTA ENCONTRAR

ALGUNO QUE NO CUMPLA. SI X FUESE K-enchado ESTE PROGRAMA BUSCARIA

E (ERNO MOMENTO, LUEGO MI PROGRAMA COMPUTA  $\text{H}(X, K)$ )

Obs: También es posible resolver este ejercicio definiendo una función no recursiva.

b) SUPONGAMOS QUE EXISTE tal K, SEA  $K_0$  ESE KENTONCES PARA TODO PROGRAMA X, X ES  $K_0$ -enchado OSEA PARA TODAENTRADA TERMINA A LO SUMO EN  $K_0$  PASOS.

ESTO ES ABSURDO. EL PROGRAMA

 $X =$  A IF  $Y = 0$  GO TO A

COMO

~~ES COMPUTABLE POR~~ COMO X INICIALMENTE EN 0, LA GUARDA SE VA A COMPAR SIEMPREOSEA ESE  $K_0$  NO VA A SER COTA SUPERIOR ABS!

NOTA Obs: Cualquier programa con al menos 1 indefinición no es K-enchado para todo K.

Y el programa  
NO VA A TERMINAR  
↑